Capítulo II : Autómatos Finitos

2.1 Autómatos Finitos Deterministas (AFD)

Definição: Um **Autómato Finito Determinista** é um quíntuplo ordenado,

$$A = (Q, \Sigma, \delta, q_0, F)$$

onde:

- Q é um conjunto finito, não vazio, de **estados**,
- Σ é um conjunto finito (alfabeto) de **símbolos de entrada**,
- $\delta : Q \times \Sigma \longrightarrow Q$ é a função (parcial) de **transição** ou de mudança **de estado**,
- $q_0 \in Q$ é o **estado inicial**,
- $F \subseteq Q$ é o conjunto dos **estados finais** ou de aceitação.

Exemplo: Sobre o alfabeto $\Sigma = \{0, 1\}$, consideremos a linguagem,

$L = \{ w \in \Sigma^* \mid$ a sequência 01 é parte de $w\}$.

Pretendemos construir um **Autómato Reconhecedor** das palavras de L. Uma dada palavra $w \in \Sigma^*$ pertence a L, se e só se, partindo do estado inicial o Autómato atingir o estado final, (de aceitação ou de reconhecimento) pela entrada de w.

A **Linguagem de um AFD** é o conjunto de todas a palavras por ele reconhecidas.

$$A = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_1\})$$

Diagrama de Transições:

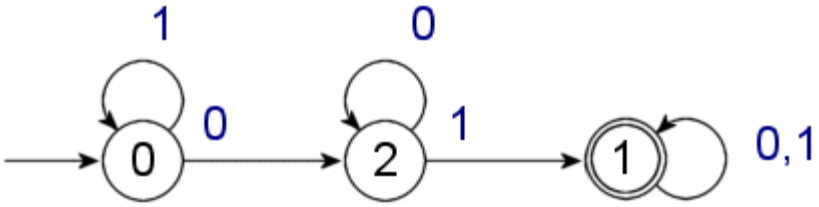


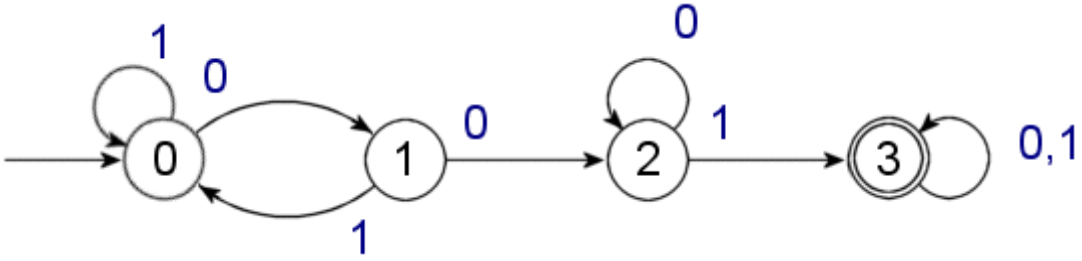
Tabela de Transições:

		0	1
$\longrightarrow$	q_0	q_2	q_0
*	q_1	q_1	q_1
	q_2	q_2	q_1

Exemplo: Também sobre o alfabeto $\Sigma = \{0, 1\}$, a linguagem,

$L = \{ w \in \Sigma^* \mid$ a sequência 001 é parte de $w\}$

tem como Autómato Reconhecedor:



Observação: A Linguagem de um AFD aparece no seu Diagrama de Transições, como o conjunto das sequências de símbolos correspondentes a todos os caminhos possíveis no grafo, desde o estado inicial até um dos estados finais.

... vai ser necessário ampliar o conceito de função de transição, por forma a aceitar sequências de símbolos.

Definição: $\delta^{\wedge} : Q \times \Sigma^{*} \longrightarrow Q$

Caso Base: $\delta^{\wedge}(q, \varepsilon) = q$

Passo Indutivo: $\forall\ w = xa,\ \delta^{\wedge}(q, w) = \delta(\delta^{\wedge}(q, x), a)$

Note-se que $\delta^{\wedge}$ representa efectivamente caminhos ao longo do grafo, isto é, se $w = a_1a_2...a_n$ e se $\delta(p_i, a_{i+1}) = p_{i+1}$ para todo o i = 0, 1, ..., n-1, então $\delta^{\wedge}(p_0, w) = p_n$.

Definição: Um AFD, $A = (Q, \Sigma, \delta, q_0, F)$ **reconhece** (ou aceita) **a palavra** $w \in \Sigma^{*}$, se e só se, $\delta^{\wedge}(q_0, w) \in F$.

Definição: **Linguagem de um AFD**, $A = (Q, \Sigma, \delta, q_0, F)$:

$$L(A) = \{\ w \in \Sigma^{*} \mid \delta^{\wedge}(q_0, w) \in F\ \}$$

Exemplo: Ainda sobre o alfabeto $\Sigma = \{0, 1\}$,

$L = \{ w \mid w$ tem um número par de 0's e de 1's $\}$.

Construção do AFD:

Basicamente trata-se de, ao longo de uma dada palavra, ir contando os 0's e os 1's (módulo 2).
Há quatro casos a considerar:

0. Os 0's e os 1's já contados, são em número par;
1. Foram já contados um número par de 0's e um número ímpar de 1's;
2. Foram já contados um número par de 1's e um número ímpar de 0's;
3. Os 0's e os 1's já contados, são em número ímpar.

Diagrama de Transições:

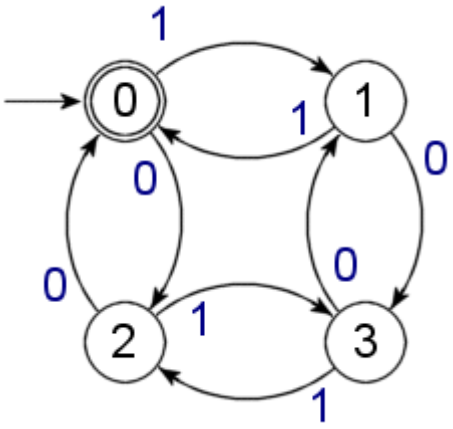


Tabela de Transições:

		0	1
$*\longrightarrow$	q_0	q_2	q_1
	q_1	q_3	q_0
	q_2	q_0	q_3
	q_3	q_1	q_2

Verificação indutiva para w = 110101:

- $\delta^{\wedge}(q_0, \varepsilon) = q_0$
- $\delta^{\wedge}(q_0, 1) = \delta(\delta^{\wedge}(q_0, \varepsilon), 1) = \delta(q_0, 1) = q_1$
- $\delta^{\wedge}(q_0, 11) = \delta(\delta^{\wedge}(q_0, 1), 1) = \delta(q_1, 1) = q_0$
- $\delta^{\wedge}(q_0, 110) = \delta(\delta^{\wedge}(q_0, 11), 0) = \delta(q_0, 0) = q_2$
- $\delta^{\wedge}(q_0, 1101) = \delta(\delta^{\wedge}(q_0, 110), 1) = \delta(q_2, 1) = q_3$
- $\delta^{\wedge}(q_0, 11010) = \delta(\delta^{\wedge}(q_0, 1101), 0) = \delta(q_3, 0) = q_1$
- $\delta^{\wedge}(q_0, 110101) = \delta(\delta^{\wedge}(q_0, 11010), 1) = \delta(q_1, 1) = q_0$

portanto $w = 110101 \in L$ □

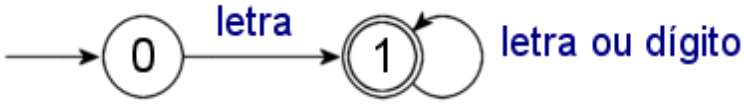
Uma aplicação: De um modo informal consideremos,

$\Sigma = \{a, b, c, ..., z\} \cup \{0, 1, 2, ..., 9\}$

$L = \{ w \in \Sigma^* \mid w$ representa um identificador$\}$,

segundo a regra habitual de construção de identificadores, nas linguagens de programação mais comuns.

AFD reconhecedor de identificadores:



Note-se que o autómato não possui a capacidade de verificar o comprimento máximo do identificador.

Exemplo: Sobre o alfabeto $\Sigma = \{0, 1\}$, a linguagem,

$$L = \{ w \in \Sigma^* \mid w \text{ é a representação binária de um múltiplo de 5} \}$$

Construção do AFD:

Em cada momento do reconhecimento, é necessário conhecer o valor do resto (da divisão por 5) do número decimal correspondente à cadeia binária já lida.

Observações:

- Sendo w a representação binária do número $\underline{n}$ (decimal), então a cadeia $w0$ representa $\underline{2\times n}$ e a cadeia $w1$ representa $\underline{2\times n+1}$.
- $(a\times b + c)$ **mod** $5 = (a\times(b$ **mod** $5) + c)$ **mod** 5.

Por exemplo:

1011 $\longrightarrow$ 11(decimal) e 11 **mod** 5 = 1 **mod** 5

10110 $\longrightarrow$ $2\times11 = 22$(decimal)

2×11 **mod** 5 $= (2\times(1$ **mod** $5))$ **mod** 5

$= 2$ **mod** 5

10111 $\longrightarrow$ $2\times11 + 1 = 23$(decimal)

$(2\times11 + 1)$ **mod** 5 $= (2\times(1$ **mod** $5) + 1)$ **mod** 5

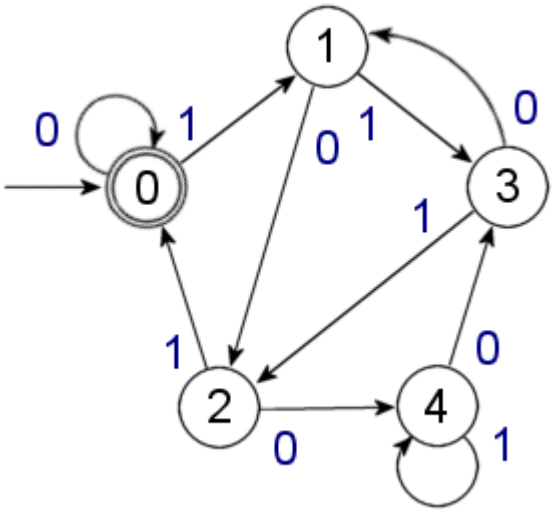
$= 3$ **mod** 5

Quantos casos há que considerar?

O AFD reconhecedor:

$$A = (\{q_0, q_1, q_2, q_3, q_4\}, \{0, 1\}, \delta, q_0, \{q_0\})$$

Diagrama de Transições:



O estado q_k significa que a cadeia binária já percorrida representa um número decimal cuja divisão por 5 tem resto igual a k.

{ Note-se que o Autómato aceita números binários começados por zero(s) e até aceita a palavra vazia }

Tabela de Transições:

		0	1
$*\longrightarrow$	q_0	q_0	q_1
	q_1	q_2	q_3
	q_2	q_4	q_0
	q_3	q_1	q_2
	q_4	q_3	q_4

Exemplo: Sobre o alfabeto $\Sigma = \{0, 1\}$, a linguagem,

$$L = \{ w \in \Sigma^+ \mid w \text{ começa por 1 e é a representação binária de um múltiplo de 5} \}$$

Basta uma pequena extensão ao Autómato anterior:

Se a palavra começar por 0, não pertence à linguagem.
Essas palavras conduzirão a um novo estado **ratoeira** (r).
A inclusão de um estado de **partida** (p) permite essa verificação, rejeitar a palavra vazia, bem como a passagem ao Autómato anterior.

O AFD reconhecedor:

$$A = (\{p, q_0, q_1, q_2, q_3, q_4, r\}, \{0, 1\}, \delta, p, \{q_0\})$$

Diagrama de Transições:

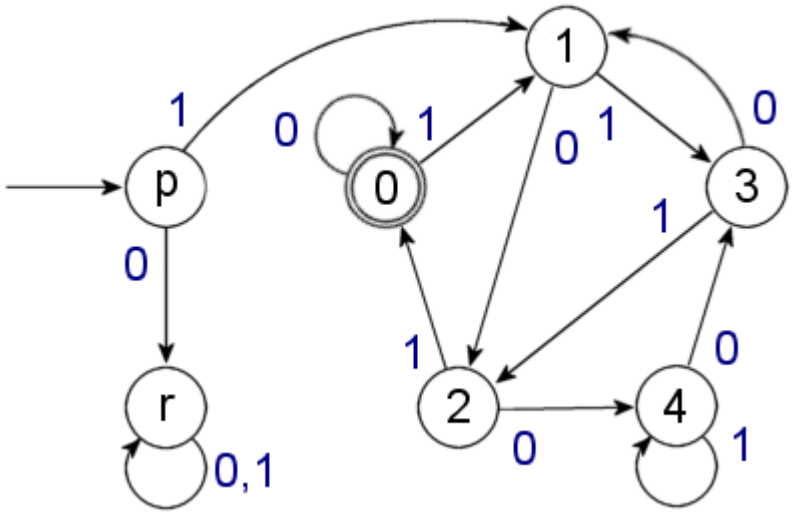
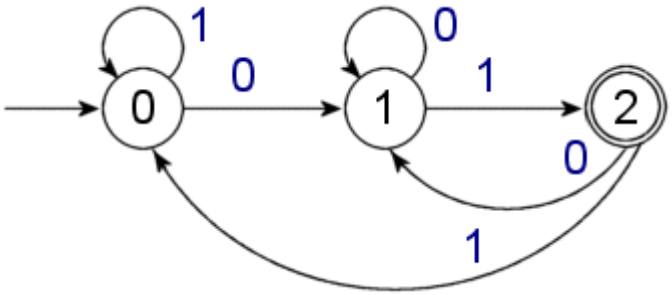


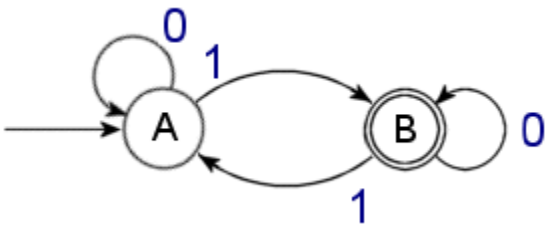
Tabela de Transições:

		0	1
⟶	p	r	q_1
*	q_0	q_0	q_1
	q_1	q_2	q_3
	q_2	q_4	q_0
	q_3	q_1	q_2
	q_4	q_3	q_4
	r	r	r

Exemplo: Um AFD reconhecedor das cadeias binárias terminadas em 01.

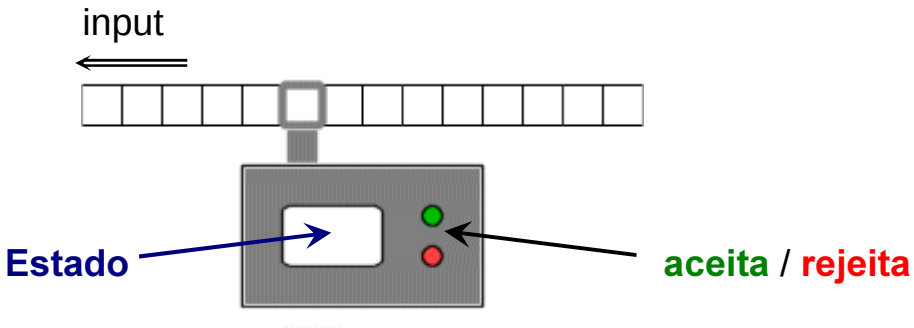


Exercício: Considere um AFD com o seguinte Tabela de Transições:



a) Descreva, informalmente, a Linguagem do Autómato.
b) Formalize a definição dessa Linguagem.
c) Demonstre, por indução, que o Autómato reconhece essa Linguagem.

"Imagem" de um Autómato Finito Determinista:



Simulação do funcionamento de um AFD:

{ Para uma dada palavra $w = \mathbf{a_1 a_2 ... a_n} \in \Sigma^*$ }

{ Sendo q ∈ Q uma variável, inicializar q ao estado inicial }

$\mathbf{q \leftarrow q_0}$

para i desde 1 até n

{ Transitar de Estado, pela entrada de a_i }

$\mathbf{q \leftarrow \delta(q, a_i)}$

{ Aqui $q = \delta^{\wedge}(q_0, w)$ }

{ A palavra *w* é reconhecida por A se e só se $\mathbf{q \in F}$ }

Implementação da simulação do funcionamento de um AFD:

- A implementação do algoritmo anterior é muito simples, na maioria das linguagens de programação.
- A função de transição $\delta : Q \times \Sigma \longrightarrow Q$ pode ser construída com base num **array**, numa estrutura **case**, ou numa combinação de ambas.

{ Programa Pascal para verificar se uma dada cadeia binária é a representação (em base 2) de um múltiplo de 5 }

```
program AFDmult5(input, output);

const   inicial = 0;
        finais = [0];
type estado = 0 .. 4;
var  q : estado;
     c : char;

function  delta(q : estado; c : char): estado;
          begin  case  q  of
                       0 :  if c = '0'
                            then delta:= 0
                            else delta:= 1;
                       1 :  if c = '0'
                            then delta:= 2
                            else delta:= 3;
                       2 :  if c = '0'
                            then delta:= 4
                            else delta:= 0;
                       3 :  if c = '0'
                            then delta:= 1
                            else delta:= 2;
                       4 :  if c = '0'
                            then delta:= 3
                            else delta:= 4
                 end
          end;

begin   write('Escreva uma cadeia binária: ');

        q:= inicial;
        while   not eoln(input) do
                begin   read(c);
                        q:= delta(q, c)
                end;

        if q in finais
        then  writeln('É a representação de um múltiplo de 5, sim senhor/a ! ')
        else  writeln('Não é nada ... ')
end.
```

- O programa anterior pode ser facilmente adaptado, por forma a descrever todo o processo de reconhecimento:

```
...
q:= inicial;
while   not eoln(input) do
        begin   read(c);
                q:= delta(q, c);
                writeln('Li um ', c,' e passei ao Estado ', q)
        end;

writeln('Acabou a cadeia. Estou no Estado ', q);
...
```

```
Escreva uma sequência binária:  11001

Li um 1 e passei ao Estado 1
Li um 1 e passei ao Estado 3
Li um 0 e passei ao Estado 1
Li um 0 e passei ao Estado 2
Li um 1 e passei ao Estado 0

Acabou a cadeia. Estou no Estado 0
```

- Também não é difícil "convencer" o mesmo AFD a calcular o valor decimal do número representado:

```
...
q:= inicial;
n := 0;
while   not eoln(input) do
        begin   read(c);
                q:= delta(q, c);
                n:= 2 * n + ord(c) - ord('0')
        end;

writeln('A cadeia binária representa o número ', n);
...
```

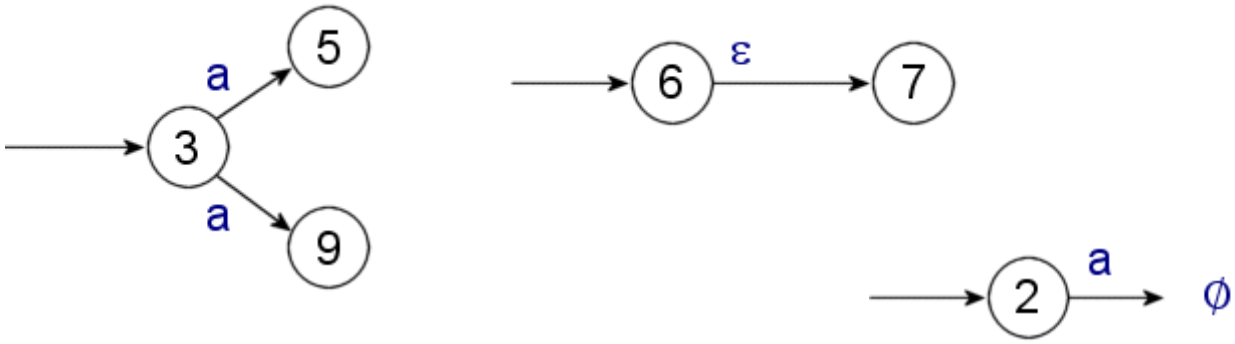
```
Escreva uma sequência binária:  11001

A cadeia binária representa o número 25
```

2.2 Autómatos Finitos Não-Deterministas (AFND)

O conceito de não-determinismo de um autómato significa que:

- ✓ A "função" de transição pode conduzir a múltiplos estados, ou mesmo a nenhum estado;
- ✓ Podem ocorrer transições por ε.

Casos de não-determinismo:



Definição: Um **Autómato Finito Não-Determinista** é um quíntuplo ordenado,

$$A = (Q, \Sigma, \delta, q_0, F)$$

onde:

- Q é um conjunto finito, não vazio, de **estados**,
- Σ é um conjunto finito (alfabeto) de **símbolos de entrada**,
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \longrightarrow \mathcal{P}(Q)$ é a **função de transição** ou de mudança **de estado**,
- $q_0 \in Q$ é o **estado inicial**,
- $F \subseteq Q$ é o conjunto dos **estados finais** ou de aceitação.

Exemplo: Um AFND reconhecedor das cadeias binárias terminadas em 01.

Diagrama de Transições:

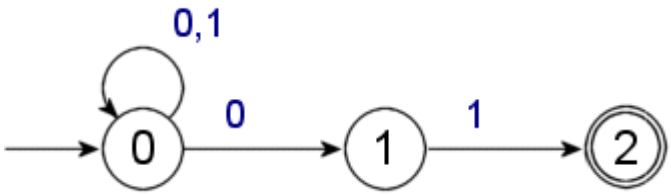
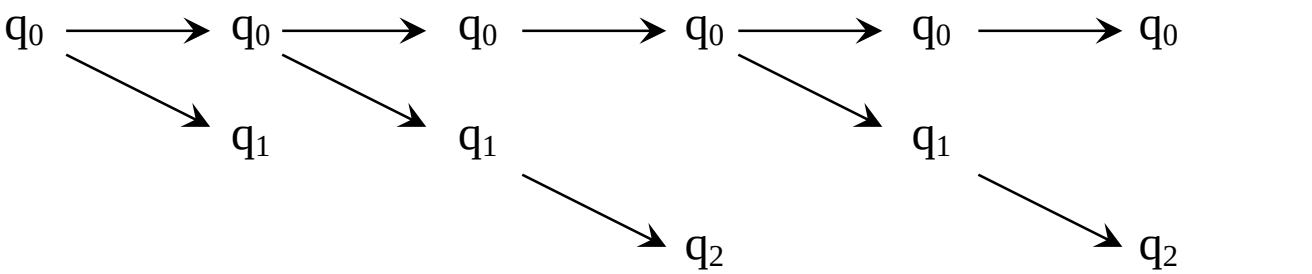


Tabela de Transições:

		0	1
→	q_0	$\{q_0, q_1\}$	$\{q_0\}$
	q_1	$\emptyset$	$\{q_2\}$
*	q_2	$\emptyset$	$\emptyset$

Consideremos a cadeia 00101 e analisemos todas as possibilidades:

0	0	1	0	1



∗(aceitação)

Esta estrutura de reconhecimento, em **árvore**, está na origem da definição do conceito de extensão da função de transição, para o caso dos AFND's.

Definição: $\delta^{\wedge} : Q \times \Sigma^* \longrightarrow \mathcal{P}(Q)$

Caso Base: $\delta^{\wedge}(q, \varepsilon) = \{q\}$

Passo Indutivo: $\forall\ w = xa$, se $\delta^{\wedge}(q, x) = \{p_1, p_2, ..., p_k\}$

e se $\bigcup_{i=1,..,k} \delta(p_i, a) = \{r_1, r_2, ..., r_m\}$

então $\delta^{\wedge}(q, w) = \{r_1, r_2, ..., r_m\}$

Por exemplo, analisemos o desenvolvimento de $\delta^{\wedge}(q_0, w)$ para w = 00101. (Notar as semelhanças com a estrutura da árvore anterior)

- $\delta^{\wedge}(q_0, \varepsilon) = \{q_0\}$
- $\delta^{\wedge}(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$
- $\delta^{\wedge}(q_0, 00) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$
- $\delta^{\wedge}(q_0, 001) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$
- $\delta^{\wedge}(q_0, 0010) = \delta(q_0, 0) \cup \delta(q_2, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$
- $\delta^{\wedge}(q_0, 00101) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

No caso de um AFND, uma palavra w é **reconhecida** quando o conjunto $\delta^{\wedge}(q_0, w)$ contem pelo menos um dos estados de aceitação. Assim:

Definição: **Linguagem de um AFND**, $A = (Q, \Sigma, \delta, q_0, F)$:

$$L(A) = \{ w \in \Sigma^* \mid \delta^{\wedge}(q_0, w) \cap F \neq \emptyset \}$$

2.3 A equivalência entre AFD's e AFND's

- ✓ Para cada AFND existe um AFD que lhe é **equivalente**, isto é, que reconhece a mesma linguagem;
- ✓ Contudo, se o AFND tiver n estados, o AFD equivalente pode ter 2^n estados.
- ✓ Mas na maioria dos casos, é possível a construção de um AFD com um número de estados próximo do AFND original.

Construção do AFD equivalente a um dado AFND:

Dado um AFND $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$,

pretende-se construir um AFD $D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$,

A construção:

- $Q_D = \mathcal{P}(Q_N)$
- $F_D = \{S \in \mathcal{P}(Q_N) \mid S \cap F_N \neq \emptyset\}$
- $\forall\, S \in \mathcal{P}(Q_N), \forall\, a \in \Sigma,$

  $\delta_D(S, a) = \bigcup_{p \in S} \delta_N(p, a)$

Teorema: $\forall\, w \in \Sigma^*$ então $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$

[Demonstração por Indução sobre $|w|$]

Corolário: $L(D) = L(N)$

Para o exemplo anterior:

$N = (Q_N, \Sigma, \delta_N, q_0, F_N) = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta_N, q_0, \{q_2\})$

	δ_N	0	1
$\longrightarrow$	q_0	$\{q_0, q_1\}$	$\{q_0\}$
	q_1	$\emptyset$	$\{q_2\}$
$*$	q_2	$\emptyset$	$\emptyset$

$D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ com $Q_D = \mathcal{P}(\{q_0, q_1, q_2\})$

analisemos todos os 2^3 estados:

	δ_D	0	1
	$\emptyset$	$\emptyset$	$\emptyset$
$\longrightarrow$	$\{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
	$\{q_1\}$	$\emptyset$	$\{q_2\}$
$*$	$\{q_2\}$	$\emptyset$	$\emptyset$
	$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$*$	$\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$*$	$\{q_1, q_2\}$	$\emptyset$	$\{q_2\}$
$*$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$

Se calcularmos todas as transições possíveis:

$\delta_D(\{q_0, q_1\}, 0) = \{q_0, q_1\}$

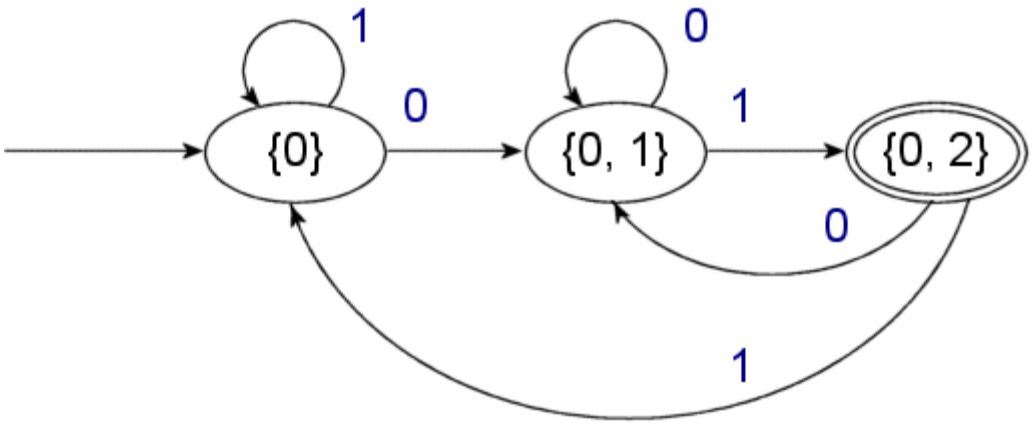
$\delta_D(\{q_0, q_1\}, 1) = \delta_N(q_0, 1) \cup \delta_N(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

$\delta_D(\{q_0, q_2\}, 0) = \delta_N(q_0, 0) \cup \delta_N(q_2, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$

$\delta_D(\{q_0, q_2\}, 1) = \delta_N(q_0, 1) \cup \delta_N(q_2, 1) = \{q_0\} \cup \emptyset = \{q_0\}$

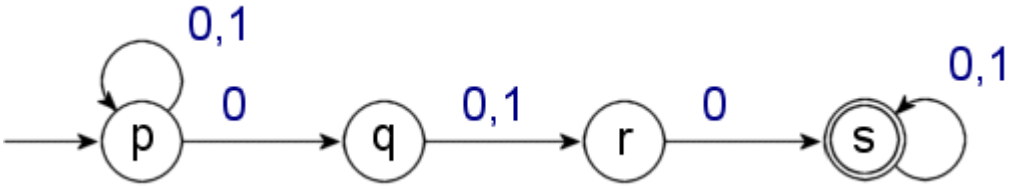
Verificamos que, dos 8 estados iniciais, apenas 3 podem ser atingidos.

Deste modo, obtemos o AFD seguinte. O número de estados é o mesmo do AFND original, mas com seis transições, em vez de quatro. (ver pág. 9)



Nota: Na maior parte dos casos, é possível construir um AFD com um número de estados semelhante ao do AFND original. {Caso médio do problema} Contudo, existem AFND´s para os quais só é possível a construção de AFD's equivalentes com 2^n estados. {O pior caso do problema}

Exemplo: Converter o seguinte AFND num AFD equivalente:



		0	1
→	p	{p, q}	{p}
	q	{r}	{r}
	r	{s}	∅
*	s	{s}	{s}

Construção do AFD:

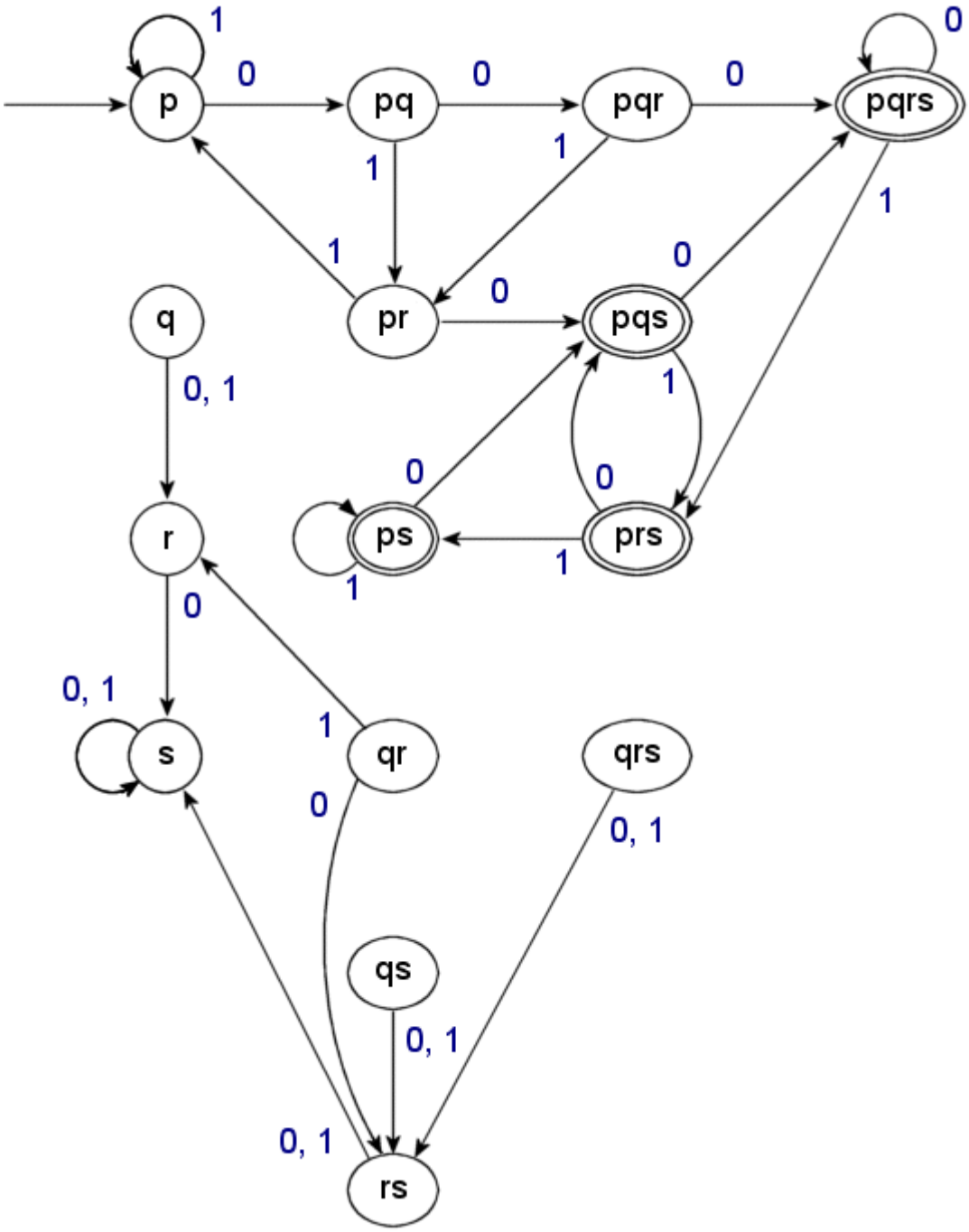
		0	1
	$\emptyset$	$\emptyset$	$\emptyset$
$\longrightarrow$	{p}	{p, q}	{p}
	{q}	{r}	{r}
	{r}	{s}	$\emptyset$
	{s}	{s}	{s}
	{p, q}	{p, q, r}	{p, r}
	{p, r}	{p, q, s}	{p}
*	{p, s}	{p, q, s}	{p, s}
	{q, r}	{r, s}	{r}
	{q, s}	{r, s}	{r, s}
	{r, s}	{s}	{s}
	{p, q, r}	{p, q, r, s}	{p, r}
*	{p, q, s}	{p, q, r, s}	{p, r, s}
*	{p, r, s}	{p, q, s}	{p, s}
	{q, r, s}	{r, s}	{r, s}
*	{p, q, r, s}	{p, q, r, s}	{p, r, s}

É simples verificar que apenas 8 dos 16 estados originais podem efectivamente ser atingidos. Assim, obtemos a tabela de transições do AFD pretendido:

		0	1
$\longrightarrow$	{p}	{p, q}	{p}
	{p, q}	{p, q, r}	{p, r}
	{p, r}	{p, q, s}	{p}
*	{p, s}	{p, q, s}	{p, s}
	{p, q, r}	{p, q, r, s}	{p, r}
*	{p, q, s}	{p, q, r, s}	{p, r, s}
*	{p, r, s}	{p, q, s}	{p, s}
*	{p, q, r, s}	{p, q, r, s}	{p, r, s}

São estados finais do AFD todos os que contém {s} = F_N.

O diagrama de transições mostra claramente a existência de dois sub-grafos disjuntos:



2.4 AFND's com Transições-ε (AFND-ε)

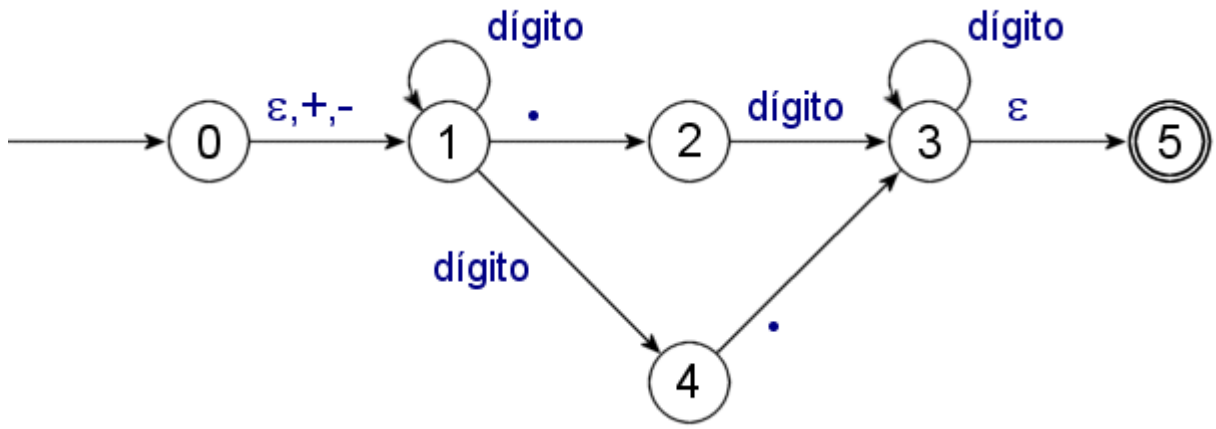
Exemplo: Um AFND-ε para reconhecer números decimais

$$A = \{\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{., +, -, 0, 1, ..., 9\}, \delta, q_0, \{q_5\}\}$$

com

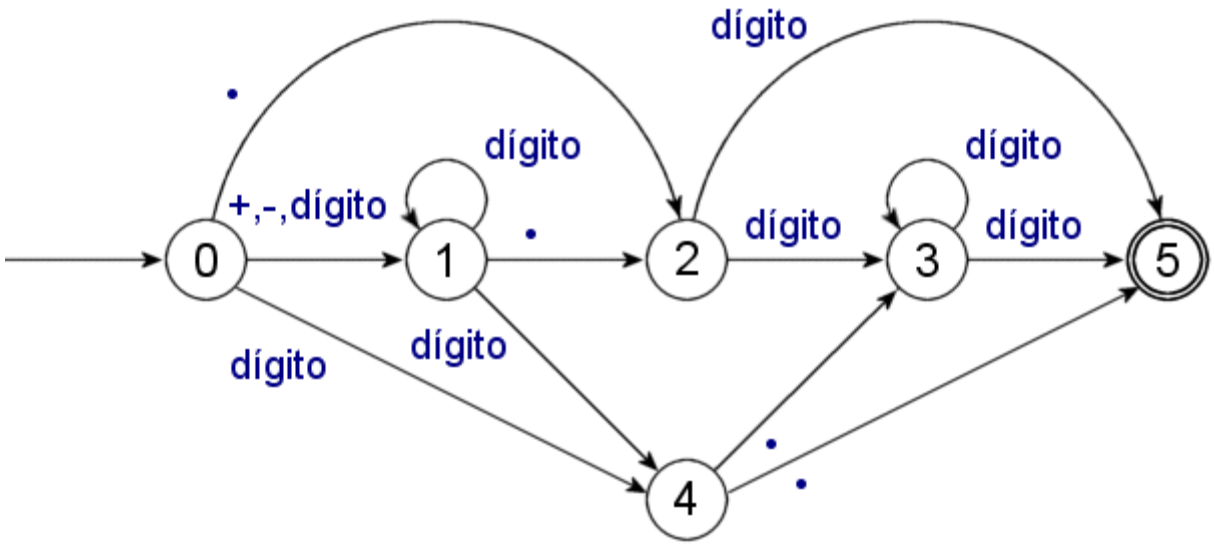
	δ	ε	+, -	.	0, 1, ..., 9
⟶	q_0	$\{q_1\}$	$\{q_1\}$	$\emptyset$	$\emptyset$
	q_1	$\emptyset$	$\emptyset$	$\{q_2\}$	$\{q_1, q_4\}$
	q_2	$\emptyset$	$\emptyset$	$\emptyset$	$\{q_3\}$
	q_3	$\{q_5\}$	$\emptyset$	$\emptyset$	$\{q_3\}$
	q_4	$\emptyset$	$\emptyset$	$\{q_3\}$	$\emptyset$
*	q_5	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

Isto é,



Note-se que são reconhecidos números decimais nas formas: 3.14
.314
314.

Nota: É sempre possível eliminar as transições-ε de um AFND-ε. Basta introduzir novas transições para "compensar o efeito" de cada transição-ε.



Contudo, a construção de um AFD equivalente a um dado AFND-ε exige a noção de fecho-ε :

Definição: fecho-ε de um estado $p \in Q$

Caso Base: $p \in$ fecho-ε(p)

Passo Indutivo: Se $q \in$ fecho-ε(p) e se existe $r \in Q$ tal que $\delta(q, \varepsilon) = r$ então $r \in$ fecho-ε(p).

Definição: fecho-ε de um conjunto de estados $C \subseteq Q$

Caso Base: se $p \in C$ então $p \in$ fecho-ε(C)

Passo Indutivo: Se $p \in$ fecho-ε(C) e se existem $p \in C$ e $q \in Q$ tais que $\delta(p, \varepsilon) = q$ então $q \in$ fecho-ε(C).

A extensão da função de transição, para o caso dos AFND-ε, também inclui o conceito de fecho-ε.

Definição: $\delta^{\wedge} : Q \times \Sigma^* \longrightarrow \mathcal{P}(Q)$

Caso Base: $\delta^{\wedge}(q, \varepsilon) = \text{fecho-}\varepsilon(q)$

Passo Indutivo: $\forall\, w = xa$, se $\delta^{\wedge}(q, x) = \{p_1, p_2, ..., p_k\}$

e se $\bigcup_{i=1, .., k} \delta(p_i, a) = \{r_1, r_2, ..., r_m\}$

então $\delta^{\wedge}(q, w) = \bigcup_{j=1, .., m} \text{fecho-}\varepsilon(r_j)$

Analisemos o desenvolvimento de $\delta^{\wedge}(q_0, w)$ para $w = 3.1$

- $\delta^{\wedge}(q_0, \varepsilon) = \text{fecho-}\varepsilon(q_0) = \{q_0, q_1\}$
- $\delta(q_0, 3) \cup \delta(q_1, 3) = \{q_1\} \cup \{q_4\} = \{q_1, q_4\}$

  $\delta^{\wedge}(q_0, 3) = \text{fecho-}\varepsilon(q_1) \cup \text{fecho-}\varepsilon(q_4) = \{q_1\} \cup \{q_4\} = \{q_1, q_4\}$
- $\delta(q_1, .) \cup \delta(q_4, .) = \{q_2\} \cup \{q_3\} = \{q_2, q_3\}$

  $\delta^{\wedge}(q_0, 3.) = \text{fecho-}\varepsilon(q_2) \cup \text{fecho-}\varepsilon(q_3) = \{q_2\} \cup \{q_3, q_5\} = \{q_2, q_3, q_5\}$
- $\delta(q_2, 1) \cup \delta(q_3, 1) \cup \delta(q_5, 1) = \{q_3\} \cup \{q_3\} \cup \emptyset = \{q_3\}$

  $\delta^{\wedge}(q_0, 3.1) = \text{fecho-}\varepsilon(q_3) = \{q_3, q_5\}$

Definição: **Linguagem de um AFND-ε**, $E = (Q, \Sigma, \delta, q_0, F)$:

$$L(E) = \{ w \in \Sigma^* \mid \delta^{\wedge}(q_0, w) \cap F \neq \emptyset \}$$

Construção do AFD equivalente a um dado AFND-ε:

Dado um AFND-ε $E = (Q_E, \Sigma, \delta_E, q_0, F_E)$,

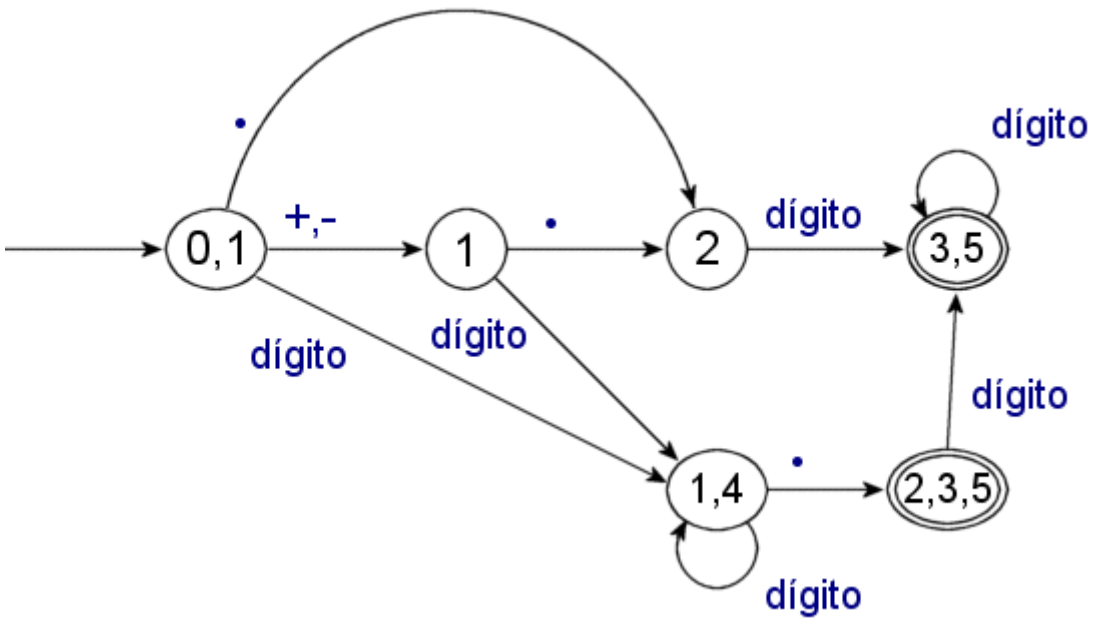
pretende-se construir um AFD $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$

A construção:

- $Q_D = \{S \in \mathcal{P}(Q_E) \mid S = \text{fecho-}\varepsilon(S)\}$
- $q_D = \text{fecho-}\varepsilon(q_0)$
- $F_D = \{S \in Q_D \mid S \cap F_E \neq \emptyset\}$
- $\forall\, S \in Q_D, \forall\, a \in \Sigma$, seja $R = \bigcup_{p \in S} \delta_E(p, a)$

  $\delta_D(S, a) = \bigcup_{r \in R} \text{fecho-}\varepsilon(r)$

Exemplo: Um AFD equivalente ao AFND-ε anterior.



Algoritmo de Conversão AFND-ε ↦ AFD:

Entrada: $E = (Q_E, \Sigma, \delta_E, q_0, F_E)$

Saída: $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$

Funções auxiliares:

vizinhos($p \in Q_E$, $a \in \Sigma$) = $\{q \in Q_E \mid \exists\, \delta_E(p, a)\}$

{estados do AFND-ε que podem ser atingidos a partir de p, por uma transição pelo símbolo a}

fecho-ε($S \subseteq Q_E$) :

T ← S
repetir R ← T
T ← R $\cup_{r \in R}$ vizinhos(r, ε)
até que T = R
fecho-ε(S) ← T

vizinhançaND($S \subseteq Q_E$, $a \in \Sigma$) = fecho-ε($\cup_{s \in S}$ vizinhos(s, a))

{estados do AFND-ε que podem ser atingidos a partir de um conjunto S de estados, por uma transição pelo símbolo a, e seguindo todas as transições por ε.}

A implementação destas três funções permite também simular o funcionamento de um AFND-ε.

Simulação do funcionamento do AFND-ε:

(Dada uma palavra $w = w_{i = 1..k}$ e sendo q_0 o estado inicial)

$q \leftarrow$ fecho-ε($\{q_0\}$)

para i desde 1 até k

    $q \leftarrow$ vizinhançaND(q, w_i)

(A palavra *w* é reconhecida se e só se $q \in F_E$)

O Algoritmo de Conversão de um AFND-ε num AFD:

$q_D \leftarrow$ fecho-ε($\{q_0\}$)

$Q_D \leftarrow Q_D \cup \{q_D\}$

para cada estado $q \in Q_D$ ainda não visitado

    visitar q

    para cada símbolo $a \in \Sigma$

        $S \leftarrow$ vizinhançaND($\{q\}$, a)

        $Q_D \leftarrow Q_D \cup S$

        inserir transição $\delta_D(q, a) = S$

para cada estado $q \in Q_D$

    se $q \in F_E$

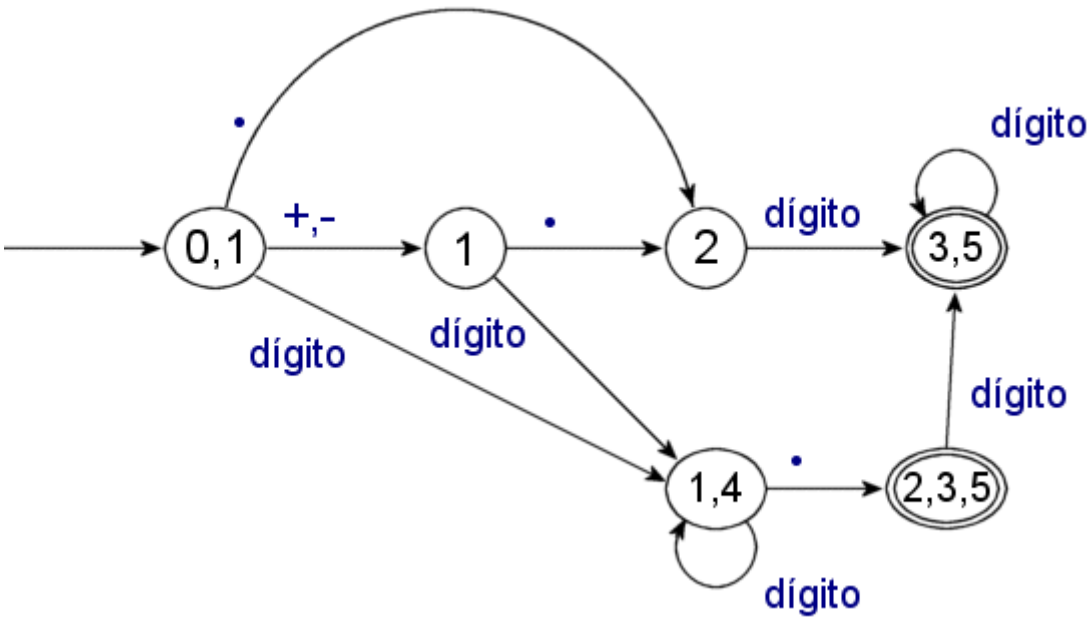
    então $F_D \leftarrow F_D \cup \{q\}$

Exemplo: $E = (Q_E, \Sigma, \delta_E, q_0, F_E)$

	δ_E	ε	+, -	.	0, 1, ..., 9
⟶	q_0	$\{q_1\}$	$\{q_1\}$	$\emptyset$	$\emptyset$
	q_1	$\emptyset$	$\emptyset$	$\{q_2\}$	$\{q_1, q_4\}$
	q_2	$\emptyset$	$\emptyset$	$\emptyset$	$\{q_3\}$
	q_3	$\{q_5\}$	$\emptyset$	$\emptyset$	$\{q_3\}$
	q_4	$\emptyset$	$\emptyset$	$\{q_3\}$	$\emptyset$
*	q_5	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

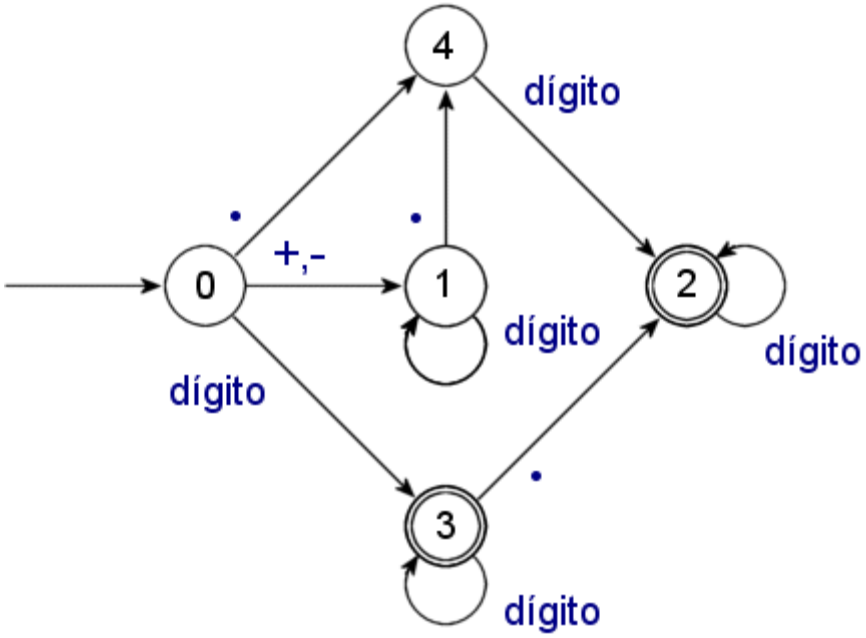
$D = (Q_D, \Sigma, \delta_D, q_D, F_D)$

		δ_D	+, -	.	0, 1, ..., 9
⟶	q_D =	$\{q_0, q_1\}$	$\{q_1\}$	$\{q_2\}$	$\{q_1, q_4\}$
		$\{q_1\}$		$\{q_2\}$	$\{q_1, q_4\}$
		$\{q_2\}$			$\{q_3, q_5\}$
		$\{q_1, q_4\}$		$\{q_2, q_3, q_5\}$	$\{q_1, q_4\}$
*		$\{q_3, q_5\}$			$\{q_3, q_5\}$
*		$\{q_2, q_3, q_5\}$			$\{q_3, q_5\}$



Notas:

- Existe sempre a possibilidade do AFD obtido ter 2^n estados.
- Não é garantido que este algoritmo produza o AFD mínimo.
- Existem algoritmos de minimização do número de estados de um AFD.
- Para o exemplo anterior, não foi difícil construir um AFD menor:



- A própria existência do Algoritmo de Conversão demonstra a **equivalência** entre AFND-ε's e AFD's.
- Contudo, não é difícil demonstrar, por Indução sobre |w|, que:

Teorema: $\forall\, w \in \Sigma^*$ então $\hat{\delta}_D(q_D, w) = \hat{\delta}_E(q_0, w)$

Corolário: $L(D) = L(E)$